

Navigation domain partitioning for interactive multiview imaging

Thomas Maugey, Ismael Daribo, Gene Cheung, and Pascal Frossard,

Abstract

Enabling users to switch interactively the viewpoint from which they are watching a static scene is an interesting functionality in 3D transmission systems. While it opens exciting perspectives towards rich multimedia applications, it requires the development of specific representation and coding techniques in order to solve the new challenges imposed by interactive navigation. In particular, the encoder must prepare a compressed media stream that is flexible enough to enable a free selection of the multiview navigation path by the different media clients. Interactivity thus brings new design constraints: the encoder is unaware of the exact decoding process, and on the other hand the decoder has to reconstruct information from incomplete subsets of data since the system can obviously not transmit images for all possible viewpoints. Furthermore, the system has to satisfy some bandwidth and storage constraints that are in general inherently conflicting, *e.g.*, a description that enables low transmission rate may lead to high storage cost. In this paper, we propose a novel multiview data representation method that tries to satisfy the above constraints. We partition the domain of multiview navigation into segments, which are described with a reference image (color and depth data) and some auxiliary information. The auxiliary information enables the client to recreate any viewpoint in the navigation segment by view synthesis. The decoder is then able to navigate independently in the segment without further data request to the server; it requests additional data only when it moves to a different sub-region. We discuss the benefits of this novel representation and further propose a method to optimize the partitioning of the navigation domain into independent segments, under bandwidth and storage constraints. Experimental results confirm the potential of the proposed representation. In particular, our system leads to similar compression performance as classical inter-view coding, while it provides the high level of flexibility that is required for interactive streaming. Our new system represents a promising alternative solution for 3D data representation in novel interactive multimedia services.

Index Terms

Multiview video coding, interactivity, data representation, navigation domain

I. INTRODUCTION

In an increasing number of multimedia applications, three dimensional data information can be used to provide an interactivity at the receiver side, where users can freely change viewpoints on their 2D displays. It enables the viewer to freely adapt his viewpoint to the scene content and provides a 3D sensation during the view navigation essentially due to the look around effect [1], [2]. The design of such an interactive system necessitates the development of new techniques in the different blocks of the 3D processing pipeline, namely acquisition [3], representation [4], coding [5], transmission [6] and rendering [7]. Solutions that are classically used for multiview video transmission [8] are no longer effective since they consider the transmission of an entire set of frames, which is not ideal for interactive systems with delay and bandwidth constraints. Interactive schemes would ideally transmit the requested images only. Hence, the challenge is to design a system that exploits the correlation between multiview images in order to satisfy the different users' requests without precise knowledge of data availability at decoder. With the classical compression techniques based on inter-image prediction with motion/disparity estimation, the problem can be solved with two naive approaches. Firstly, if the server is able to store all the possible encoding prediction paths between the achievable views, the user can receive only the required frames (with a prediction corresponding to its navigation path) at low bitrates. Alternatively, it is also possible to consider a real-time encoding (and thus real-time inter-image prediction) [9] depending on the user position. However these two solutions do not scale with the number of users and rely on infinite storage and delay-free communication assumptions, which are not realistic in practical settings. The challenge for realizing an interactive multiview system for static 3D scene transmission is thus twofold: i) decrease the storage size without penalizing too much the transmission rate and ii) anticipate user requests and prepare data accordingly. This has to be done considering of the complete system, from the data capture to the view rendering blocks, including representation and coding strategies.

A few solutions in the literature try to optimize the trade-off between storage, bandwidth and interactive experience. A first category of methods optimize switching between captured views only. In other words, they adapt the structure of the inter-view predictions in order to provide interactivity at a moderate cost. Some of these methods are inspired by the techniques that have been developed to provide interactivity for monoview video. For example the concept of SP/SI frames [10] is adapted in [11] for view-switching. Other works propose to modify the prediction structure between the frames [12], [13] by predicting

T. Maugey and P. Frossard are with École Polytechnique Fédérale de Lausanne (EPFL), Signal Processing Laboratory - LTS4, CH-1015 Lausanne, Switzerland. Email: thomas.maugey@epfl.ch, pascal.frossard@epfl.ch

I. Daribo and G. Cheung are with National Institute of Informatics, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan 101-8430. E-mail: daribo@nii.ac.jp, cheung@nii.ac.jp

the user position with the help of Kalman filtering. The authors in [14] propose to store multiple encodings on the server and to adapt the transmission to the user position. This is however very costly in terms of storage. In [15], [16], the multi-view sequence is encoded with a GoGOP structure, which corresponds to a set of GOPs (Group of Pictures). The limitation of such methods is a fixed encoding structure that cannot be easily adapted to different configurations. In [17], the problem is formulated so that the proposed prediction structure reaches an optimal trade-off between storage and expected streaming rate. The possible types of frames are intra frames and predicted frames (with the storage of different motion vectors and residuals). Some other techniques [18], [19], [20] rely on the idea of combining distributed source coding and inter-view prediction for effective multiview switching. They propose an extension of the view switching methods in a monoview framework [21]. Unfortunately, all of these solutions remain limited since they restrict the navigation to a small subset of views (the captured ones, generally not numerous), which results in abrupt, unnatural view-switching experience. Moreover, they cannot directly be extended to a system that provides smooth navigation all over the scene at the receiver (with higher number of achievable viewpoints).

A second category of methods try to offer free viewpoint navigation by considering a higher number of achievable views at the receiver. It could be obtained by simply increasing the number of captured views, which is not feasible in practice and not efficient in terms of redundancy in the representation. Some solutions [22] extend the previously mentioned techniques by introducing virtual view synthesis at the decoder. However, they remain inefficient since the obtained virtual view quality is low and the user navigation capacity is still limited. Other methods introduce high redundancy in the scene description by using a light field representation [23], [24]. They sample the navigation domain very finely, concatenate all the images and finally model the light rays of the scene. The view rendering performed at the receiver side with such light fields has a better quality and enables quite smooth navigations. However, the navigation is constrained and the data representation does not achieve good compression performance. In general, all the solutions that multiply the number of possible views have *inherent redundancies in the representation*, which results in an inefficient streaming system.

It is important to note that all of these methods do not use an end-to-end system design approach. For example, while optimizing the coding techniques, almost none of the above works consider the constraints of the data rendering step. It results in data blocks with strong dependencies which are not necessarily optimized for interactive navigation. In this work, we build on [25] and propose a radically new design that is supported by a flexible data representation method for static 3D scenes. It also includes new methods for data compression and for view rendering at the receiver. The proposed solution achieves a high quality free-viewpoint navigation experience and limits data redundancies in the system. Instead of optimizing data representation for a small set of predefined viewpoints, we rather consider that free viewpoint navigation is described by a navigation domain with all the possible virtual camera locations. The navigation domain (ND) is divided into sub-domains called *navigation segments*, which are transmitted to the decoder upon request. Upon reception of data from a navigation segment, the decoder can independently create any virtual view in this sub-domain without further request to the server. It provides a high navigation capability at the receiver. But it also implies a complete change in the data representation in order to limit storage and bandwidth costs. Each navigation segment is then represented with a reference frame and some auxiliary information. The auxiliary information carries in a compact form the innovation inherent to new viewpoints and permits to synthesize any view in the navigation segment with approximately the same quality. Then, we propose to optimize the partitioning of the navigation domain under rate and storage constraints. We finally illustrate the performance of our system on several datasets with different configurations. We observe that the proposed data representation achieves good compression performance with high flexibility for interactive user navigation. This new method offers a promising alternative solution for the design of 3D systems with new modes of interactions and rich quality of experience.

The paper is organized as follows. In Sec. II, we introduce the principles of our system. Then, we expose in Sec. III our solution to optimize the partitioning of the navigation domain. Finally, in Sec. IV, we present different simulations results that validate the proposed approach.

II. INTERACTIVE MULTIVIEW NAVIGATION

A. System overview

In an interactive system, the user is able to freely navigate between a large set of viewpoints, in order to observe a static scene from different virtual camera positions. It generally means that the user has to communicate with a server and to request data that permits reconstruction and rendering of the desired virtual views on a 2D display. Let us consider a navigation domain constituted by a set of viewpoints. Our system relies on a novel data representation method that goes beyond the common image-based representation and rather considers the global navigation domain as union of different navigation segments. So let us consider that the navigation domain is divided in N navigation segments, each of them being coded in a single data D_i (see Fig. 1). We will see later that D_i is globally represented in the form of one reference image and some auxiliary information. We further consider that a server stores all the D_i , with a consecutive storage cost of $\sum_{i=1}^N |D_i|$ ($|D_i|$ being the size in bits of the data D_i). A user navigating between the viewpoints is regularly transmitting its position to the server. If a user in a navigation segment i comes close to a border with another navigation segment j , the server transmits the data D_j to the user, which increments the reception rate cost by $|D_j|$. We see that, if the partitioning of the navigation domain is dense (*i.e.*, N is

high), on the one hand, the segment size $|D_i|$ decreases, but on the other hand, the user requests more often to the server. On the contrary, if N is low, the number of requests to the server decreases, but the user has to receive heavy D_i . We clearly see that N should be determined carefully, taking into account the bandwidth and storage constraints.

We notice that the communication between server and user is quite simple. It has to deal with data transmission only close to the borders of the navigation segments. Hence, this can be generalized to multiple users scenario easily. However, if, at the end, the number of users becomes very high, one can consider a multiple servers system, but this is out of the scope of the paper.

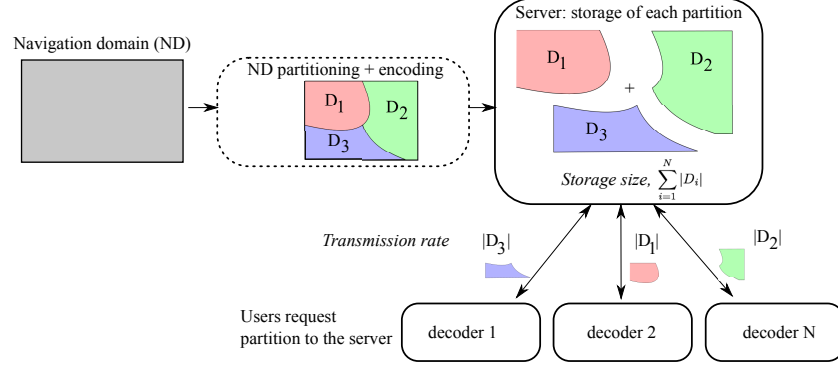


Fig. 1. Navigation domain is first partitioned. Each navigation segment is encoded and stored on a server. Users interact with the server to request the navigation segments needed for the navigation.

B. Navigation with 2D images

We provide now a formal description of the interactive multiview framework that we propose in this paper. We consider a system that captures and transmits data of a 3D *scene* S to clients that can reconstruct 2D images of the scene for navigation, *i.e.*, view-switching along certain directions. The scene S is described as a countable set of random variables s_i taking values in C^3 , where C is the set of possible color values (*e.g.*, $[0, 255]^3$)¹. Each of these random variables can be seen as a voxel in the 3D space [26]. The decoder reconstructs observations of the scene at different viewpoints. These observations are 2D *images* that correspond to finite sets of N random variables x_i taking their values in C^3 . The observation of the 3D scene from one particular viewpoint gives an image X that is obtained with a *projection function* associated to X . Since the depth information is known, we rather define the back projection function which associates a pixel of an image to a 3D point in the scene:

$$f_X : X \rightarrow S$$

$$x \rightarrow s = f_X(x).$$

This projection function depends on the distance between objects and camera plane (*i.e.*, depth) and on the extrinsic and intrinsic parameters of the camera [1], [27], [28], [29]. In this work, we assume that each pixel in X maps to a single voxel in 3D space S , and reciprocally, each voxel in S maps to maximum one pixel in X (in other words, f_X is a bijection of X in $f_X(X) \subset S$). This assumption is correct as long as the 3D scene is sampled at a sufficiently high resolution, which is the scenario that we consider in the following. Not all the elements of S can be seen from one viewpoint. We call $S_X = f_X(X)$ the finite subset of S whose elements are mapped to elements of X . This is the set of elements of S that are *visible* in X . It naturally depends on the viewpoint. Our objective is to deliver enough information to the decoder, such that it can reconstruct different images of the scene. At the same time, the images from different viewpoints have a lot of redundancy. Ideally, to reconstruct an image X' knowing the image X , decoder should be sent only the non-redundant information. We define it as the *innovation* of X with respect to X' : $I_{X,X'} = S_X \setminus S_{X'}$ (see Fig. 2). This innovation is due to two classical causes in view switching. First, *disocclusions* represent the most complex source of innovation. They are due to pixels that are hidden by a foreground object or that are out of the camera range in the first view and become visible in the second view. The disocclusions are generally not considered at the encoder in the literature. Existing schemes consider that they can be approximated by inpainting [30], [31] or partially recovered via projection from another view [27]. Although, the performance of inpainting techniques is improving, there still exists a problem with new objects or with frame consistency (especially when neighboring frames are not available in interactive systems). These two issues should be considered at the encoder and data to resolve disocclusions should also be sent to the decoder. We propose below a new data representation method for interactive multiview navigation based on these two important ideas.

¹In this work we make the Lambertian hypothesis, *i.e.*, we assume that a voxel reflects the same color even when viewed from different viewpoints.

Second, innovation can also be generated by some new elements that appear due to a variation in the object resolution, *i.e.*, when an object is growing from one viewpoint to another one. In other words, two consecutive pixels representing the same object in X could map to two non-consecutive ones in X' (even if they still describe the same object), and let the pixel(s) between unmapped to other pixels. This is due to the bijection assumption introduced above. However, we have chosen to restrict our study to the handling of disocclusions and we assume that these resolution-variation missing pixels are recovered by a simple interpolation of the neighboring available pixels (of the same object). This assumption remains reasonable if we consider a navigation without large forward displacements. This is actually what is classically done in the view synthesis papers [27], [28]. Therefore, in the experiments, we will only consider navigation trajectories that remains at a similar distance from the scene.

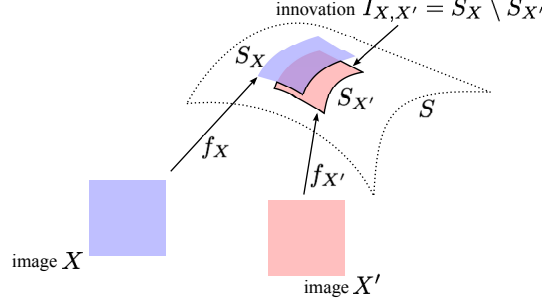


Fig. 2. Illustration of visibility of scene elements in the images X and X' . The innovation of X' with respect to X is represented with the black line.

C. Navigation domains

We define the new concept of *navigation domain* as a contiguous region that gathers different viewpoints of the 3D scene S , with each of these viewpoints being available to the users to be reconstructed as a 2D image (see Fig. 3). This is an alternative to the classical image-based representation used in the literature, where a scene is represented by a set of captured views [32]. In our framework, the concept of captured camera or virtual view does not exist anymore, in the sense that all the images of the navigation domain are equivalent. We denote by $c(X) \in \mathbb{R}^p$ the camera parameter vector associated to the image X :

$$c(X) = c_X = \underbrace{[t_x \ t_y \ t_z]}_{\text{translation}} \underbrace{[\theta_x \ \theta_y \ \theta_z]}_{\text{rotation}}^T.$$

From these parameters, we define the navigation domain as a continuous and bounded domain $\mathcal{C} \in \mathbb{R}^p$. We associate to \mathcal{C} the dual image navigation domain: $\mathcal{X} = \{X | c_X \in \mathcal{C}\}$. In the following, a navigation domain (ND) refers to both the set \mathcal{C} and its dual definition.

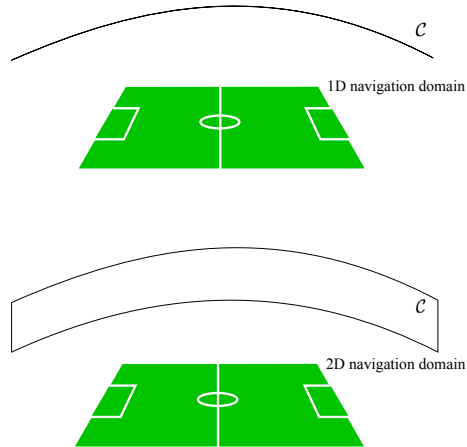


Fig. 3. The navigation domain can be 1D or 2D, and is defined by the set of camera parameters \mathcal{C} .

The new concept of navigation domains permits us to have a general formulation of the view switching problem. Naturally it also leads to novel data representation methods. The main idea of our novel approach is first to divide the navigation domain

into non-overlapping partitions, \mathcal{X}_i , called *navigation segment*. In other words, we have $\mathcal{X} = \bigcup_i \mathcal{X}_i$ with $\mathcal{X}_i \cap \mathcal{X}_j = \emptyset$ for all i and j . Then, we represent all the views in one segment with one signal, which is used at decoder for user view switching.

Each navigation segment is first described by one *reference image*, called Y . This image is important as it is used for the baseline reconstruction of all images in the segment. We thus denote the navigation segment as $\mathcal{X}(Y)$, which represents the set of images that are reconstructed from a reference Y at the decoder. The reference image completely determines the rest of the navigation segments under some consideration about the geometry of the scene and the camera positions, as explained later. We therefore use the simplified notation $\mathcal{X}(Y)$ for uniquely describing a navigation segment. The part of the scene visible from the reference image Y is called $S_Y = f_Y(Y)$ (it is illustrated in plain red lines in Fig. 4). At the decoder, a user-selected image X is reconstructed using depth-image-based rendering techniques (DIBR [33]) that project the frame Y onto X , *i.e.*, the decoder builds $f_X^{-1}(S_Y)$. The decoder is thus missing the elements of information in $X \setminus f_X^{-1}(S_Y)$ for each view X of the navigation segment. Some of these missing elements in different X 's map to the same voxel. This is why we merge the different innovations and we define the *global segment innovation* as

$$\Phi = \bigcup_{X \in \mathcal{X}(Y)} S_X \setminus S_Y. \quad (1)$$

It correspond to a global information that is missing in Y to recover the whole navigation segment. It is represented in blue dashed lines in Fig. 4. The objective is to design an auxiliary information, for transmitting this set Φ to the decoder, that is very light and takes the coded form $\varphi = h(\Phi)$.

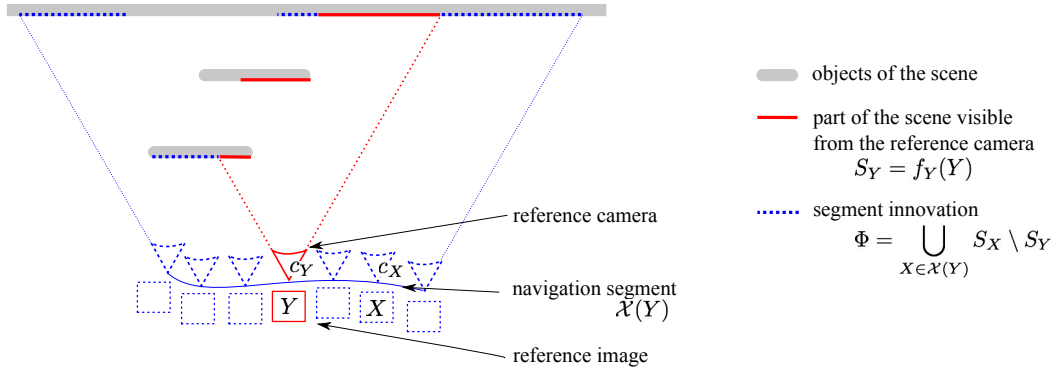


Fig. 4. Illustration of the concept of navigation segment for a simple scene (represented from top) with one background (vertical plane) and two foreground objects (vertical rectangles).

Equipped with our new data representation method, we can finally describe our communication system in detail. We assume that a server stores the different navigation segments that compose the whole navigation domain. This storage has a general cost Γ . At the receiver, a user navigates among the views, chooses to build a 2D image X at a viewpoint described by parameters $c_X \in \mathcal{C}$. The only constraint in the navigation is that the user cannot choose randomly his viewpoint, he has to switch smoothly to the neighboring images. We define a *distance* δ between two camera parameter vectors c and c' as $\delta : (c, c') \rightarrow \delta(c, c')$. This distance is computed between the camera parameters vectors. We can consider different distances whether we want to emphasize rotation or translation. We note also $\delta : (X, X') \rightarrow \delta(c_X, c_{X'})$ the dual distance between two images X and X' . Since \mathcal{C} is a continuous set, we define Δ as the *navigation step*, which corresponds to the distance between two different images chosen at consecutive instant. We assume that the user can send his position in the navigation domain every N_T frame². Once the user sends its position, the server transmits all the navigation segments where the user might need in the next N_T instants. We define the *navigation ball* as the set of achievable viewpoints in the next N_T instants from the viewpoint X as:

$$B(X, N_T \Delta) = \{X' \in \mathcal{X} | \delta(X, X') < N_T \Delta\}. \quad (2)$$

In other words, the server sends all $\mathcal{X}(Y_i)$ that verify $\mathcal{X}(Y_i) \cap B(X, N_T \Delta) \neq \emptyset$. Finally, the user navigation depends on the *a priori* view popularity distribution, $p(X)$ ($X \in \mathcal{X}$), which corresponds to a dense probability distribution over the views. It describes the relative popularity of the viewpoints (we have $\int_{X \in \mathcal{X}} p(X) = 1$) as in practice not all viewpoints have the same probability to be reconstructed at decoder.

D. Data coding

We now describe the data coding method that is used in this paper. This method is not fully optimized in terms of compression and is certainly not exclusive; it however nicely fits the design choices described above. Recall that each navigation segment is composed of one reference image Y and some auxiliary information $\varphi = h(\Phi)$, where Φ is the segment innovation. First,

²If f is the frame rate, N_T can be expressed in seconds by dividing the value expressed in number of frames by f

the images Y (color and depth data) are coded and stored using classical intra frame codecs such as H.264/AVC Intra [8]. We use such reference images to generate all the other views of the navigation segment $\mathcal{X}(Y)$ *via* view synthesis. As explained before, the set of frames $X \in \mathcal{X}(Y) \setminus Y$ contains a certain innovation Φ that represents the global novelty of the views in the navigation segment with respect to Y . In practice, we estimate this set as follows (it is also illustrated in Fig. 5). We first project the image Y in the 3D scene using depth information (in other words, we compute S_Y). Then, we project every frame X from the segment $\mathcal{X}(Y)$ in the 3D scene using depth information (*i.e.*, we compute S_X). In our representation, each pixel is associated with a voxel in the 3D space and some voxels are shared by two images. In practice, Φ is the union of voxels visible in views in $\mathcal{X}(Y)$ but not visible in Y . In order to avoid redundancies, the voxels shared by different views in $\mathcal{X}(Y)$ are only represented once in Φ . In the following, we will use the concept of size of Φ , which simply corresponds to the number of voxels in the set Φ , denoted as $|\Phi|$. We will see that this size has a strong impact on the rate of the auxiliary information, denoted by $|\varphi|$.

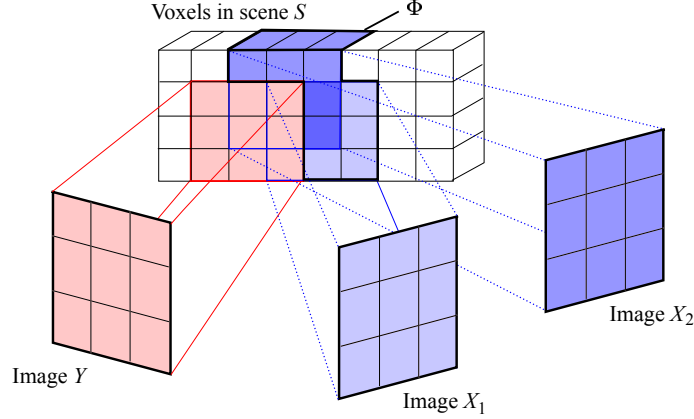


Fig. 5. Example of Φ construction, when images Y , X_1 and X_2 are projected to the 3D scene. In that example, Φ is made of 9 voxels, thus $|\Phi| = 9$.

We still have to encode the auxiliary information for reducing its size. The encoding function (*i.e.*, the function $\varphi = h(\Phi)$) consists in building a quantized version of DCT blocks from the auxiliary information image, which can gather the segment innovation Φ when cameras are aligned. The innovation segment image is then divided into small pixel blocks that are DCT transformed and quantized. The bitstream is then encoded with a classical arithmetic coder. If the navigation domain is more complex, this approach can be extended to the layered depth image (LDI [34]) format. In that case, auxiliary information in each layer can be DCT transformed and quantized.

At the decoder we exploit the auxiliary information in a reconstruction strategy that is based on the Criminisi's inpainting algorithm [30]. It is made of two steps. During the first one, the algorithm chooses the missing image patch that has the highest priority based on image gradient considerations. The second step fills the missing information by using a similar patch from the reconstructed parts of the image. We modify Criminisi's inpainting algorithm by introducing in this second step a distance between the candidate patch and the auxiliary information. The hole filling technique thus chooses a patch that corresponds to the auxiliary information $h(\Phi)$. This auxiliary information is deduced from the φ signal and projected to the patch position using depth information. Finally, it is important to note that the design of the auxiliary information coding technique does not depend on the decoder. Also, the reconstruction technique is independent of the type of hash information that is transmitted.

III. OPTIMAL PARTITIONING OF THE NAVIGATION DOMAIN

A. Constrained partitioning

The new data representation proposed above raises an important question, namely the effective design of the navigation segments. We show here how the partitioning can be optimized under rate and storage constraints. Recall first that the objective is to represent the navigation domain as the union of N_V navigation segments. Thus, we have $\mathcal{X} = \bigcup_{i=1}^{N_V} \mathcal{X}(Y_i)$, where $\mathcal{X}(Y_i)$ is the set of images reconstructed from the reference image Y_i and the associated auxiliary information.

Let us study in a first step a simple case, which will permit to define the new concept of similarity between two frames. We assume that N_V is given and that the references images Y_i 's are already fixed. A naive way of defining a navigation segment $\mathcal{X}(Y)$ consists in decomposing the ND based on the distance between cameras:

$$\forall i \in [1, N_V], \quad \mathcal{X}(Y_i) = \{Y_i\} \cup \{X \in \mathcal{X} | \forall j \neq i, \delta(X, Y_i) \leq \delta(X, Y_j)\}. \quad (3)$$

This definition leads to equidistant reference image distribution over the navigation domain as shown in Fig. 7(a). However, this definition takes into account neither the scene nor the notion of innovation between the images. This is why we define the

geometrical similarity γ between two images as (for sake of conciseness, geometrical similarity will be replaced by similarity in the rest of the paper):

$$\gamma: (X, X') \rightarrow \gamma(X, X') = |S_X \cap S_{X'}|. \quad (4)$$

This similarity definition lays the foundations of a new kind of correlation between images, when two images share a set of identical pixels but also contain sets of independent pixels. In other words, instead of considering a model where the correlation between two images is an error all over the pixels, as it is classically adopted in image coding, we use a model where, two pixels of two different images are either equal or totally independent. This new kind of correlation between images is measured by the similarity function of Eq. (4). This leads to a novel partitioning defined as:

$$\forall i \in [1, N_V], \quad \mathcal{X}(Y_i) = \{Y_i\} \cup \{X \in \mathcal{X} | \forall j \neq i, \gamma(X, Y_i) \geq \gamma(X, Y_j)\}. \quad (5)$$

It reflects the quantity of innovation between two images and leads to non-equidistant partitioning. Typically, the navigation segments are smaller if the similarity varies quickly with the distance between cameras (Fig. 7(b)). To illustrate the fact that similarity is not linearly dependent on the distance between cameras, we present a simple experiment in Fig. 6. For the *Ballet* sequence [35], we build a navigation domain made of 100 equidistant viewpoints (in the sense of the camera parameters). For two reference images (index 1 in Fig. 6(a) and 50 in Fig. 6(b)) we calculate their similarity with all the other frames of the navigation domain. The similarity is expressed here between 0 and 1 and corresponds to a percentage of common pixels³, *i.e.*, of pixels that are associated to the same voxels in the 3D scene. We can actually see that the evolution of the similarity function is not linear with the view index nor with the distance, *i.e.*, the non-linear (plain red lines) interpolation fits better the similarity function than the linear one (dashed black line).

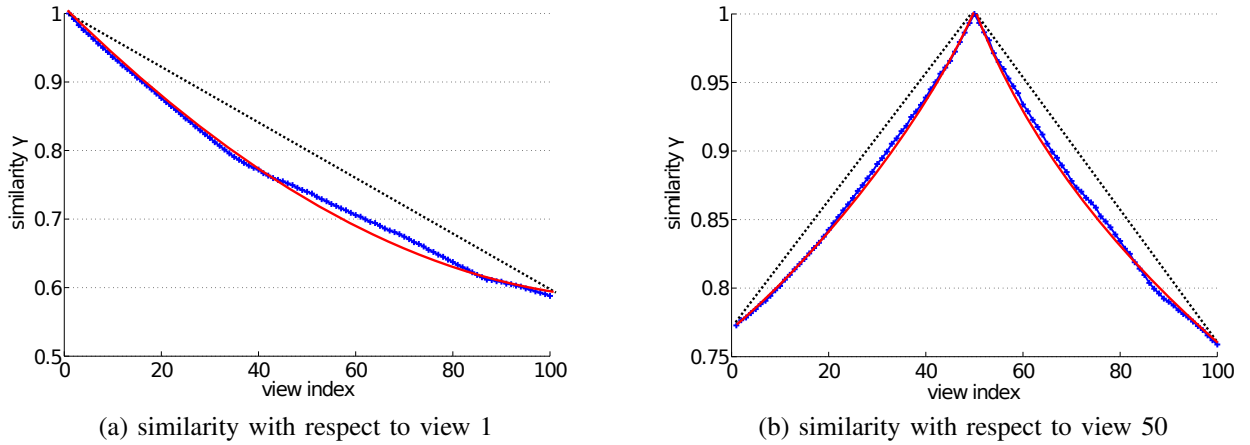


Fig. 6. Similarity evolution (blue crosses) in function of the view index of a navigation domain, in which the images are equidistant. Black dashed line corresponds to linear interpolation between the extreme values while red plain curve is a non-linear interpolation of the curve which obviously fits better the similarity evolution.



Fig. 7. Visual difference between the two distance-based and similarity-based partitioning for 1D navigation domain.

Equipped with this new fundamental notion of similarity, let us further develop the concept of optimal partitioning. It is obtained by fixing the right number of reference views N_V^* and choosing the proper reference images Y_i^* . Partitioning is optimized with respect to a storage size I , which corresponds to the total cost of storing all the navigation segments, and with respect to a rate R , which corresponds to an average transmission cost. We assume that the navigation step Δ is fixed. We further assume that if a user starts its navigation on a reference frame, the navigation segment is sufficiently big to enable independent navigation during N_T frames without transmission of another navigation segment. The optimal partitioning problem can be posed as:

³The similarity is normally defined as a number of voxels in the 3D space, however, for this test, we have chosen to divide it by the size of the image in order to obtain a value between 0 and 1, which makes the interpretation easier.

$$(N_V^*, \{Y_i^*\}) = \arg \min_{(N_V, \{Y_i\})} R(N_V, \{Y_i\}) \quad \text{under the constraint that } \Gamma(N_V, \{Y_i\}) \leq \Gamma_{\max}. \quad (6)$$

We rewrite the above as an unconstrained problem with help of a Lagrangian multiplier λ as

$$(N_V^*, \{Y_i^*\}) = \arg \min_{(N_V, \{Y_i\})} R(N_V, \{Y_i\}) + \lambda \Gamma(N_V, \{Y_i\}) \quad (7)$$

with $S(N_V, \{Y_i\}) = \sum_{i=1}^{N_V} |Y_i| + |\varphi_i|$ and Γ_{\max} is a storage constraint. The storage Γ depends on both the size of the reference frame $|Y_i|$ and the auxiliary information $|\varphi_i|$, with $\varphi = h(\Phi)$ being the coding function for each navigation segment. The transmission rate R corresponds to the expected size of the information to be sent after each request and measures the average size of the navigation segment. Note that, formally, it differs from the notion of bandwidth, expressed in bit per second. It depends on navigation models or view popularity and is written as:

$$\begin{aligned} R(N_V, \{Y_i\}) &= \sum_{i=1}^{N_V} P(\mathcal{X}(Y_i)) R(\mathcal{X}(Y_i)) \\ &= \sum_{i=1}^{N_V} P(\mathcal{X}(Y_i)) (|Y_i| + |\varphi_i|) \end{aligned} \quad (8)$$

where $P(\mathcal{X}(Y_i)) = \int_{X \in \mathcal{X}(Y)} p(X)$ corresponds to the probability that the user navigates in segment $\mathcal{X}(Y)$, as proposed in Sec. II. We assume that the relation between the coding rate and the auxiliary information has been characterized a priori, as it depends on the coding method. We propose a method to solve this optimization problem in the next section.

B. Optimization method

We first assume that N_V^* is given. We notice that the optimization problem of Eq. (7) is similar to a problem of vector quantization [36]. It consists in dividing the vector space in different partitions, represented by codewords that are chosen to minimize the reconstruction distortion under a rate constraint. Here we want to find a partitioning of the navigation domain that minimizes the rate under storage constraints, while the quality of the reconstruction is not affected. In Lloyd algorithm [36] for vector quantization, the positions of the codeword determine the quantization cells; similarly the position of Y_i determines $\mathcal{X}(Y_i)$ in our partitioning problem. More precisely, in an ideal case, the definition of the navigation segment becomes $\mathcal{X}(Y_i) = \arg \min_{\{X\}} (|Y| + |\varphi|)$. We consider that it can be achieved from Eq. (5), which builds the segment with the elements that have a higher similarity with the reference frame than with the reference frames of the other navigation segments. Intuitively, the similarity is a correlation measure between two frames, which permits to form navigation segments when the reference frames are given. The problem now consists in selecting the reference frames Y_i 's. We consider a simple iterative algorithm that performs three steps:

- step 1: initialize the reference frames Y_i 's
- step 2: derive optimal navigation segments given the reference frames Y_i 's, based on frame similarity criteria in Eq. (5)
- step 3: refine the reference frame in each navigation segment in order to minimize storage and rate costs in Eq. (7).

The algorithm then proceeds iteratively with an alternation between steps 2 and 3. It ends when the refinement in step 3 does not provide a significant storage and rate gain. Convergence is guaranteed, because the same objective $R + \lambda \Gamma$ is minimized in steps 2 and 3, and the objective function is bounded from below.

It remains now to define the number of segments, *i.e.*, the value N_V^* . For that purpose, we need to define a maximum number of navigation segment M . It corresponds to the case where all the segments have the minimum acceptable size, *i.e.*, the size of the navigation ball defined in Eq (2). We write M as follows :

$$M = \frac{\text{size}(ND)}{\text{size}(B(X, N_T \Delta))}.$$

The *size* of $C \subset \mathcal{C}$ is defined as $\int_{x \in C} \mathbb{1}(x) dx$ (where $\mathbb{1}$ is the classical indicator function). It results that N_V^* lies between 1 and M . We then determine N_V^* as

$$N_V^* = \arg \min_{1 \leq N_V \leq M} \Gamma + \mu R_{\max} \quad (9)$$

where R_{\max} is the maximum navigation segment size that the user receives per request during navigation. The parameter μ corresponds to a parameter that permits to regulate the relative importance of rate with respect to storage cost. To solve Eq. (9), we calculate the storage and rates values for N_V taking each integer value between 1 and M :

- $\Gamma = N_V |\bar{Y}| + N_V |\bar{\varphi}|$, where $|\bar{Y}|$ and $|\bar{\varphi}|$ are rough estimations of the average reference frame rate and reference auxiliary information rate. They are deduced from the coding strategy adopted for $\varphi = h(\Phi)$.

- $R_{\max} = |\bar{Y}| + |\bar{\varphi}|$.

Finally we have the optimal value of the number of navigation segments by exhaustive search, as

$$N_V^* = \arg \min_{1 \leq N_V \leq M} (N_V + \mu)|\bar{Y}| + (N_V + \mu)|\bar{\varphi}|. \quad (10)$$

IV. EXPERIMENTS

A. Setup

Our novel interactive system is tested on two well-known multiview sequences provided by Microsoft research [35]⁴, namely *Ballet* and *Breakdancer*. Each of these sequences is composed of eight texture and depth videos and their associated intrinsic and extrinsic parameters. From these multiview images, we build a navigation domain that is composed of 120 viewpoints (texture and depth), as illustrated in Fig. 8. We also build a 2D navigation domain which consists of 5 distinctive rows of these 120 horizontal alignment of viewpoints. In order to create the viewpoints that are not present in the original sequences, we use view synthesis techniques [1]. All of the images (camera images and synthetic images) form our input dataset; they are considered as original images, and can be chosen as reference frames by the partitioning algorithm. Finally, we index images in this set of equidistant viewpoints from 1 to 120 for the 1D navigation domain, and from (1, 1) to (5, 120) for the 2D one.

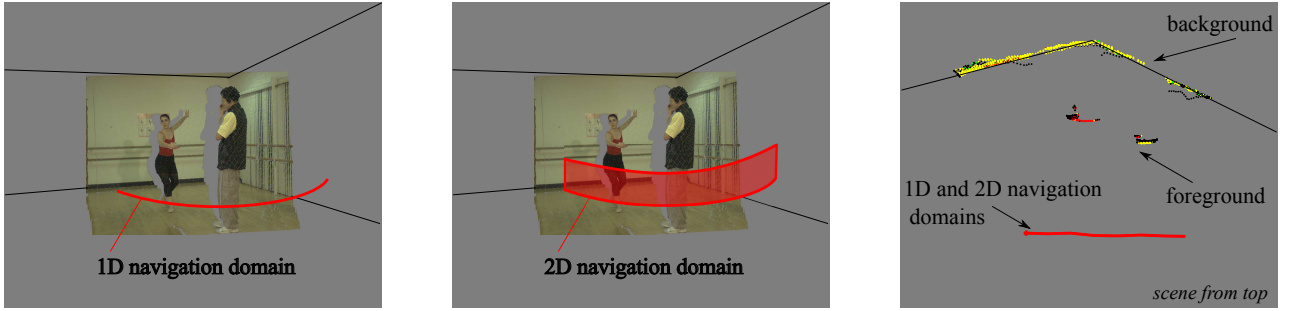


Fig. 8. Illustration of 1D and 2D navigations domain used in the experiments for *Ballet* sequence.

B. Disocclusion filling based on coded auxiliary information

We first study the performance of disocclusion filling based on auxiliary information. This permits to validate the reconstruction strategy that is at the core of our new data representation method. An example of reconstruction with the proposed inpainting technique is shown in Fig. 9(a). It is obtained by first projecting a reference view Y on a virtual view X (see Fig. 9(a), the disocclusions are in white). Secondly the disoccluded regions are reconstructed using the classical Criminisi's algorithm (Fig. 9(b)) and our guided inpainting method (Fig. 9(c)). We can see that the reconstructed quality obtained with our method is very satisfying. Moreover, the side information used for this illustrative example is not heavy in terms of bitrate, as shown in Fig. 10. In these experiments, we measure the rate (at different quantization steps) of the following schemes: a single view transmission, two views coded jointly, and our proposed representation (one view and the auxiliary information φ). We observe that the rate of our representation method is much smaller than the rate needed for sending two reference views for synthesis.

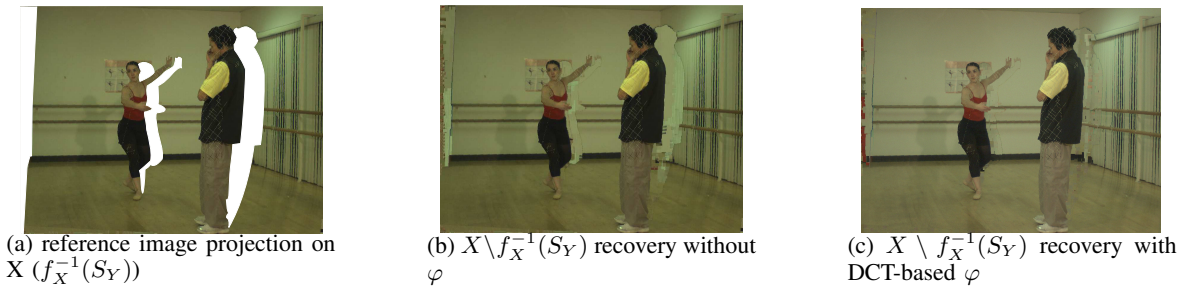


Fig. 9. Visual results for reconstruction of view 2 in *Ballet* using view 1 as reference image. We compare classical inpainting method (b) and proposed guided inpainting method (c).

In the next experiments shown in Sections IV-C to IV-E we have chosen to present the result in terms of number of voxels in the segment innovation, $|\Phi|$, instead of rate and the storage costs. This choice is motivated by the following reasons. In Eq. (7), the rate and storage costs depend on $|\varphi|$, which is the size (in kbits) of the auxiliary information. This auxiliary

⁴Since we are considering the navigation in a static image, we only consider the temporal instant 1.

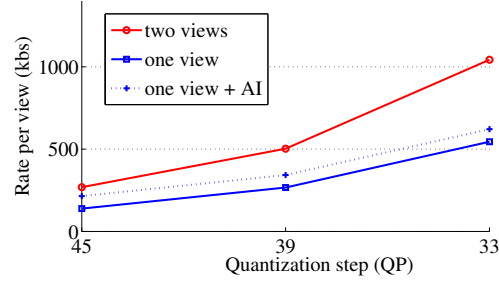


Fig. 10. Rate comparison between different representation for the navigation segment: single view, two views, one reference image + auxiliary information.

information is a compressed version of the segment innovation Φ . Intuitively, the quantity of information in φ is increasing when the number of elements in the segment innovation Φ is increasing. We can also observe that the increase is linear with the auxiliary information design presented above (see Fig. 11). By showing the results in terms of $|\Phi|$, we make the choice of presenting general results that can be adapted to any kind of encoding function h . However, in order to present also more concrete performance of our solution we show in Sec. IV-F some rate and storage results obtained with a practical implementation of the system (based on a auxiliary information constructed using DCT coefficients as introduced above).

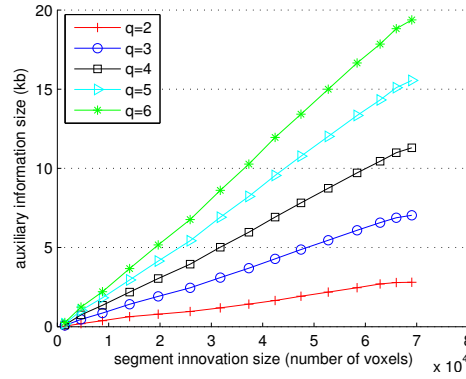


Fig. 11. Illustration of auxiliary information size $|\varphi|$ evolution as a function of the number of voxels in the segment innovation $|\Phi|$, for the practical implementation of DCT based auxiliary information design (q corresponds to the number of bits used to describe each DC coefficient).

C. Influence of reference view

We now study the influence of the position of a reference view within a navigation segment. One of the strengths of the proposed representation is to avoid the differentiation between captured and synthesized views. Every frame is considered with the same importance, which gives a new degree of freedom in navigation performance optimization via proper selection of the reference view Y_i . We evaluate the impact of the position of Y_i on the size of the segment innovation Φ . We illustrate in Fig. 12 and 13 the typical evolution of Φ as a function of Y_i , in 1D and 2D navigation domains. More precisely, we fix the navigation segments and vary the position of $\{Y_i\}$. For each position, we calculate $|\Phi|$ as explained in Sec. II-D. We see that the shape of these curves is approximately convex, but non regular and non symmetric. The size of the auxiliary information clearly depends on the scene content. We see that the position of Y_i has a strong impact on the size of the segment innovation, and therefore on the rate of the encoded auxiliary information, since $|\varphi|$ depends on $|\Phi|$. We see that the size $|\Phi|$ can even double, depending on the Y position.

D. Optimal partitioning

We discuss now the results of the optimized partitioning algorithm and its effect on the size of the segment innovation Φ . We assume here that the number of partitions N_V is predetermined. We run the algorithm with databases introduced in Sec. IV-A. Since the shape of the criterion function in our optimization problem is not completely convex, one needs to be careful in the initialization of the algorithm in order to avoid local minima. We put the initial reference frames at equidistant positions (in terms of geometrical similarity), which has been shown experimentally to be a good initial solution. In Fig. 14 and 15, we show the performance of our algorithm in the partitioning of a 1D navigation domain. In each of these figures we show (a) the evolution of the partitioning and (b) the evolution of the segment innovation $|\Phi|$ through the successive steps

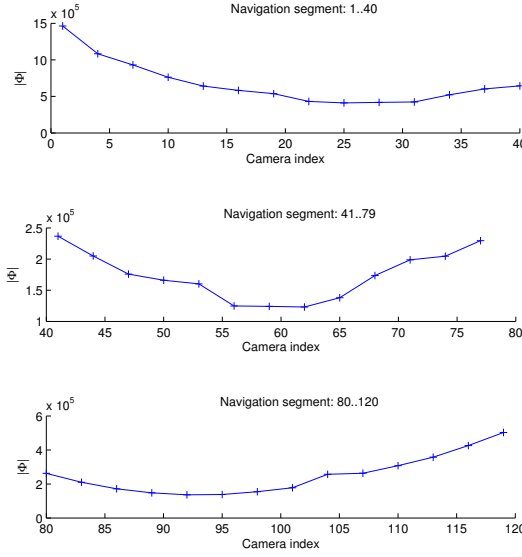


Fig. 12. Size of the segment innovation Φ (measured in number of voxels) as a function of the reference frame position (expressed in terms of camera index within the general navigation domain) for a 1D navigation domain.

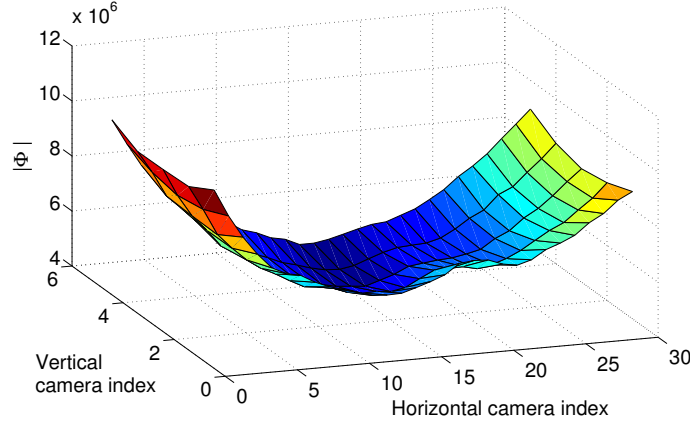


Fig. 13. Size of the segment innovation Φ (measured in number of voxels) as a function of the reference frame (expressed in terms of camera index within the general navigation domain) position for a 2D navigation domain.

of the partitioning algorithm. We see that, for $N_V = 2$ and $N_V = 3$, the total segment innovation decreases and the size of each segment innovation Φ converges to a similar value. We also remark that the algorithm converges in a small number of steps. We show in Fig. 16 and 17 similar results for the partitioning of a 2D navigation domain. We illustrate the final 2D partitioning and the evolution of the segment innovation size along the steps of the algorithm. We can see that the algorithm converges quickly and manages to decrease the size of the neighborhood innovation Φ . It is interesting to notice that the resulting partitioning does not correspond to an equidistant distribution of the reference frames (in terms of camera parameter distance). The proposed algorithm takes into account the scene content in the definition of the navigation segments.

E. Optimal number of navigation segments

We now study how the system determines the appropriate number of navigation segments. The optimal number of navigation segments N_V^* is determined by minimizing the criterion given in Eq. (10). We show in Fig. 18 the shape of this criterion function with different values of the relative weight factor μ . For these tests, we have considered that the coding function h is linear (i.e., φ increases linearly with Φ), as it is experimentally obtained in Fig. 11. Then, for each value of N_V , we roughly estimate the storage and maximum rate costs for the *Ballet* sequence. We see that we obtain different optimal number of navigation segments N_V^* depending on the parameter μ that trades off storage and rate costs. We see that, if μ is large (i.e., more importance is given to the rate cost), the algorithm selects a high number of navigation segments. On the contrary, if the storage cost has more importance, the system prefers a small number of navigation segments. The parameter μ thus regulates the importance of the storage cost with respect to rate costs. This parameter is determined during the design of the system and depends on the system constraints and network delivery conditions.

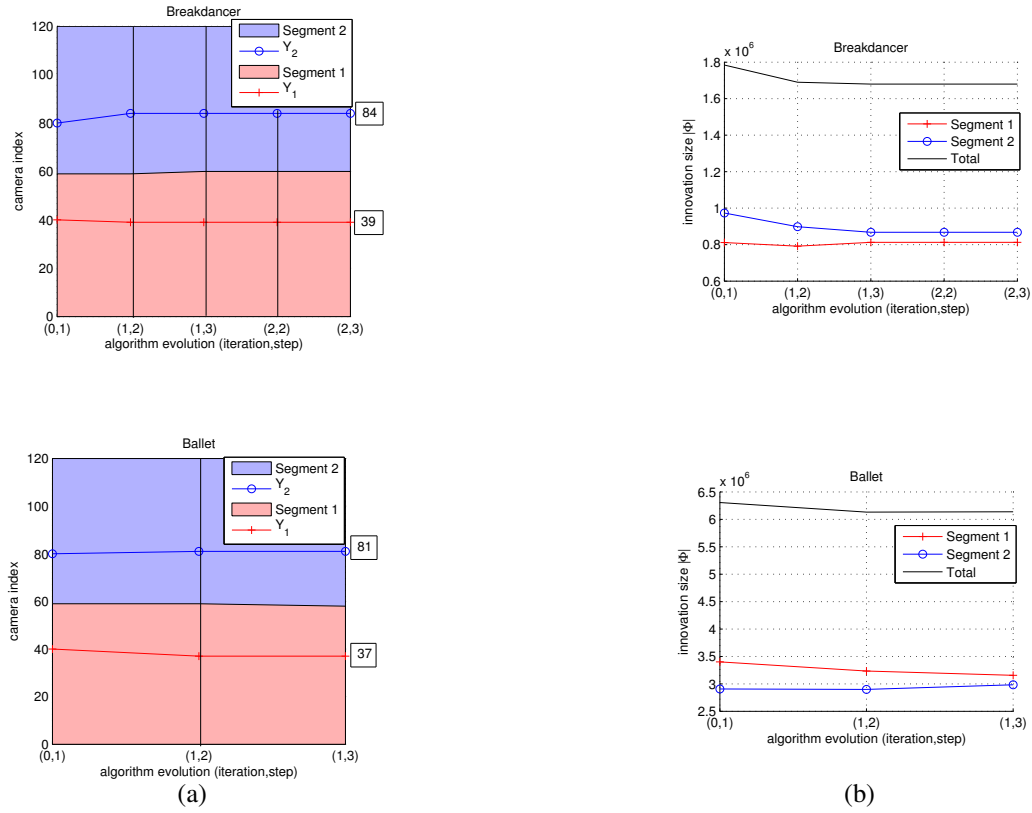


Fig. 14. Partitioning results for two 1D navigation segments when the initial reference frames are set at positions 40 and 80 (camera indexes). On the left, the evolution of the partitioning is illustrated as a function of algorithm evolution expressed as (a, b) , where a is the iteration and b is the step. On the right, the corresponding $|\Phi|$ evolution is plotted.

F. Rate and storage performance

So far, we have mainly presented partitioning results in terms of the size of the segment innovation $|\Phi|$ which is directly related to the rate and storage costs. We now present results that illustrate the performance of our algorithm in terms of rate values. We encode the auxiliary information with a quantized DCT representation as introduced in Sec. II-D, which leads to a linear relation between the rate and the size $|\Phi|$ (as illustrated in Fig. 11). We first model a possible navigation path for a user navigation of a duration of 100s. The obtained path is represented in Fig. 19(a). For this navigation path, we simulate the communication of the client with the server and we plot the evolution of the bit rate at client in Fig. 19(b), with the initial partitioning and the one optimized with our algorithm. Here, the initial partitioning corresponds to the regular distribution of reference frames at the initialization of the optimization algorithm. We further plot the cumulative rate of the navigation instance as a function of time for the same two partitioning solutions. We see that the rate per second significantly decreases with the optimal partitioning; similarly the cumulative rate after 100s of navigation is also smaller when the partitioning is optimal. Similar results have been obtained for different navigation paths and different values of the number of navigation segments N_V . To generalize these results, we have averaged the cumulative rate after 100s for 100 navigation paths, and for different values of N_T . We show the results in Fig. 20. We can see from all these representative results that the partitioning optimization leads to significant rate cost reductions. This validates our partitioning optimization solution.

Finally, in order to figure out the efficiency of the proposed representation method, we compare the storage cost of the proposed system with a naive solution. The latter consists in jointly compressing the 8 captured images (with JMVM [37]), and in interpolating the other frames with bidirectional DIBR, as it is classically done in the literature. We use two different partitioning solutions ($N_V = 2$ and $N_V = 3$) and we use the DCT-based auxiliary information coding explained in Sec. II-D. The storage cost calculated contains the transmission of the reference image (color and depth) and the auxiliary information (for our solution). The quality is estimated over a representative sample of images (8, 23, 38, 53, 68, 83, 98, 113 and the 8 reference views). The results are shown in Fig. 21 where we see that the proposed representation obtains similar compression performance as the 8 views in JMVM without auxiliary information. However, since the use of JMVM compression is not realistic in an interactive scenario, the proposed representation certainly results in very promising performance even if the compression of the auxiliary information is not fully optimized yet.

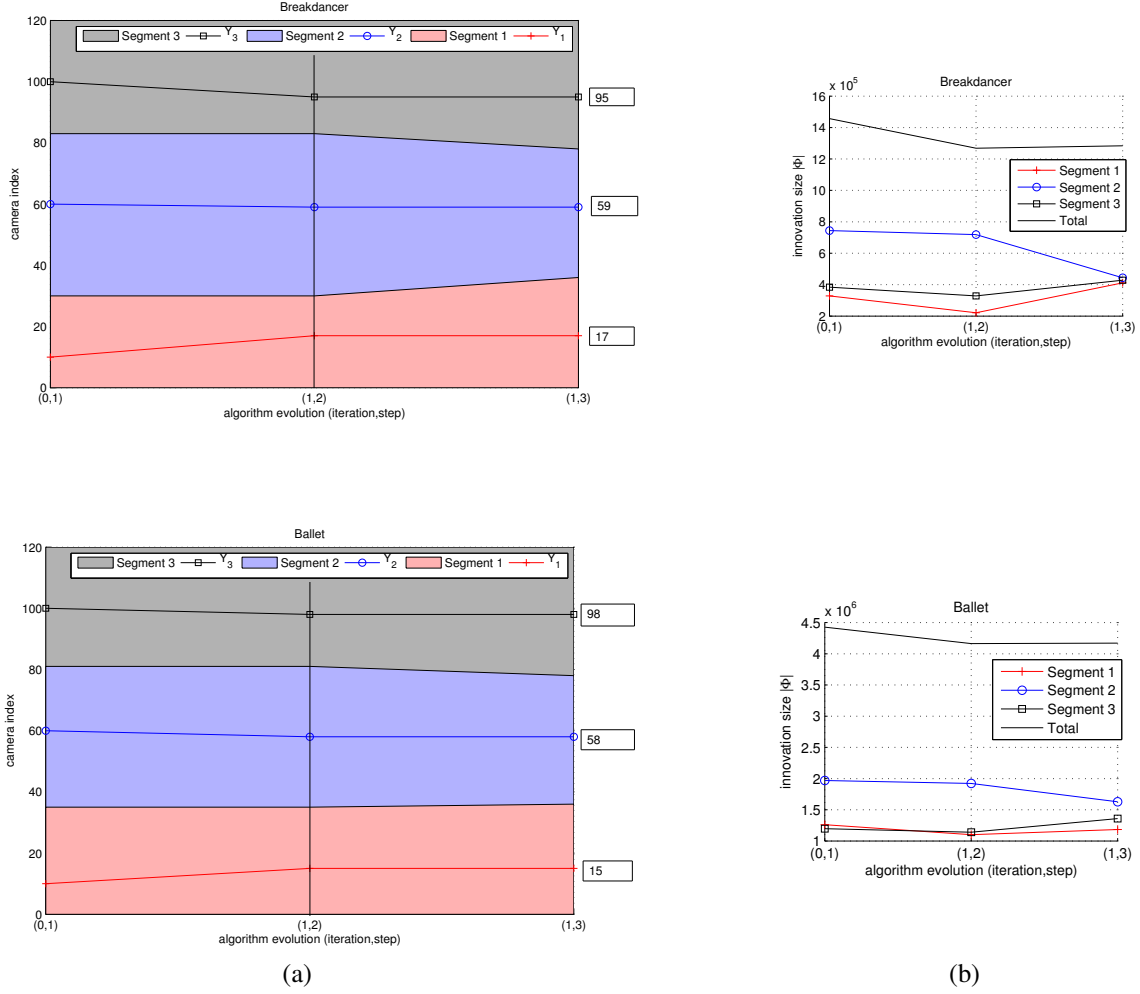


Fig. 15. Partitioning results for three 1D navigation segments when the initial reference frames are set at positions 10, 60 and 100 (camera indexes). On the left, the evolution of the partitioning is illustrated as a function of the algorithm evolution expressed as (a, b) , where a is the iteration and b is the step. On the right, the corresponding $|\Phi|$ evolution is plotted.

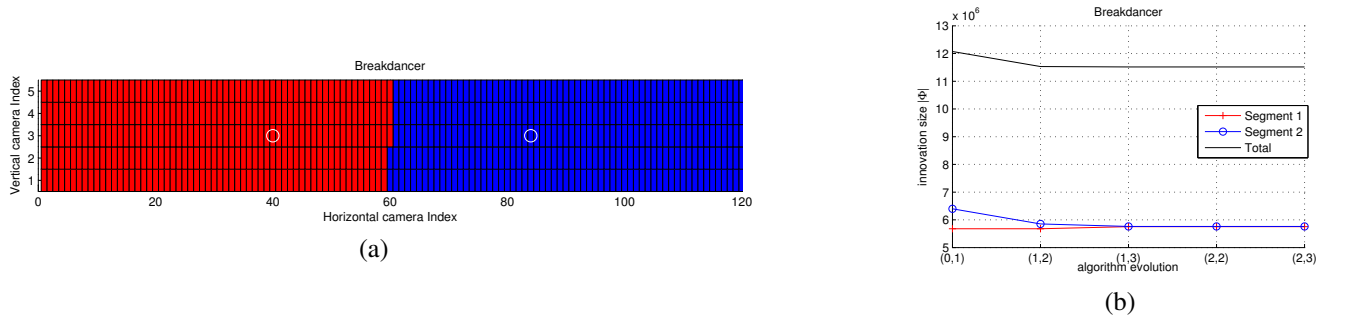
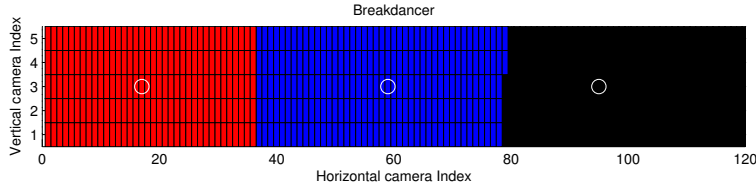


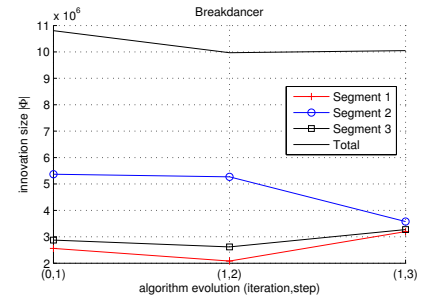
Fig. 16. 2D partitioning results for two navigation segments with initial reference frames at positions (3, 30) and (3, 60) (camera indexes). On the left, the final partitioning is illustrated; on the right, the evolution of the innovation size $|\Phi|$ is plotted as a function of the algorithm evolution expressed as (a, b) , where a is the iteration and b is the step.

V. CONCLUSION

In this paper, we propose a novel data representation method for interactive multiview imaging. It is based on the notion of navigation domain which is optimally splitting into several navigation segments. Each of these navigation segments is described with one reference image and auxiliary information, which enables a high quality user navigation at the receiver. In addition of this novel concept, we have proposed an effective solution to partition the navigation and find the best position for reference images. Experimental results show that the viewing experience of the user is significantly improved with a reasonable rate and storage cost. The comparison with the existing standards are encouraging and show the potential of such approach for future systems. Future work will mainly focus on the extension of this idea to temporal aspects in order to enable efficient interactive



(a)



(b)

Fig. 17. 2D partitioning results for three navigation segments with initial reference frames at positions (3, 10), (3, 60) and (3, 100) (camera indexes). On the left, the final partitioning is illustrated; on the right, the evolution of the innovation size $|\Phi|$ is plotted as a function of the algorithm evolution expressed as (a, b) , where a is the iteration and b is the step.

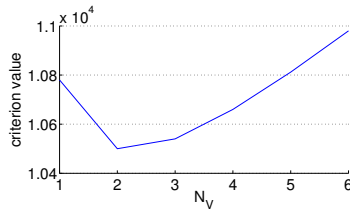
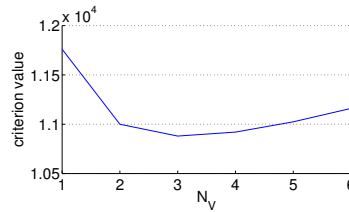
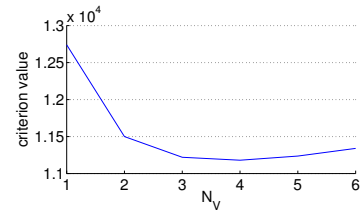
(a) $\mu = 0.1$ (b) $\mu = 0.2$ (c) $\mu = 0.3$

Fig. 18. Optimal number of navigation segments N_V^* for different values of relative weight-factor μ .

multiview video viewing.

ACKNOWLEDGMENT

This work has been partly support by the Swiss National Science Foundation, under grant 200021-126894

REFERENCES

- [1] K. Müller, P. Merkle, and T. Wiegand, "3d video representation using depth maps," *Proc. IEEE*, vol. 99, no. 4, pp. 643–656, Apr. 2011.
- [2] A. Smolic and P. Kauff, "Interactive 3D video representation and coding technologies," *Proc. IEEE*, vol. 93, no. 1, pp. 98–110, Jan. 2005.
- [3] P. Benzie, J. Watson, P. Surman, I. Rakkolainen, K. Hopf, H. Urey, V. Sainov, and C. von Kopylow, "A survey of 3DTV displays: Techniques and technologies," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, no. 11, pp. 1647 – 1658, Nov. 2007.
- [4] A. Alatan, Y. Yemez, U. Gündükbay, X. Zabulis, K. Müller, C. Erdem, C. Weigel, and A. Smolic, "Scene representation technologies for 3dtv - a survey," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, pp. 1587–1605, 2007.
- [5] A. Smolic, K. Müller, N. Stefaniski, J. Ostermann, A. Gotchev, G. Akar, G. Triantafyllidis, and A. Koz, "Coding algorithms for 3dtv - a survey," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, pp. 1606–1621, 2007.
- [6] A. Vetro, A. Tourapis, K. Müller, and T. Chen, "3d-tv content storage and transmission," *IEEE Trans. on Broadcasting*, vol. 57, pp. 384–394, 2011.
- [7] N. Holliman, N. Dodgson, G. Favalora, and L. Pocket, "Three-dimensional displays: a review and applications analysis," *IEEE Trans. on Broadcasting*, vol. 57, no. 2, pp. 362–371, Jun. 2011.
- [8] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
- [9] J. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *Proc. ACM Int. Conf. on Multimedia*, Singapore, 2005, pp. 161–170.
- [10] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 13, no. 7, pp. 637–644, Jul. 2003.
- [11] Y. Chen, Y. Wang, K. Ugur, M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standards for 3d video services," *EURASIP J. on Adv. in Sign. Proc.*, vol. 2009, pp. 1–13, 2009.
- [12] E. Kurutepe, M. Civanlar, and A. Tekalp, "Client-driven selective streaming of multiview video for interactive 3DTV," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, no. 11, pp. 1558–1565, Nov. 2007.
- [13] A. Tekalp, E. Kurutepe, and M. Civanlar, "3DTV over IP: end-to-end streaming of multiview videos," *IEEE Signal Processing Magazine*, no. 6, pp. 77–87, Nov. 2007.
- [14] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, "RD-optimized interactive streaming of multiview video with multiple encodings," *Journal on Visual Commun. and Image Repr.*, vol. 21, no. 5-6, pp. 1–10, Jul. 2010.
- [15] H. Kimata, M. Kitahara, K. Kamikura, and Y. Yashima, "Free-viewpoint video communication using multi-view video coding," *NTT Technical Review*, vol. 2, no. 8, pp. 21–26, Aug. 2004.
- [16] S. Shimizu, M. Kitahara, H. Kimata, K. Kamikura, and Y. Yashima, "View scalable multiview video coding using 3-d warping with depth map," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, no. 11, pp. 1485–1495, Nov. 2007.
- [17] G. Cheung and N. Ortega, A. and Cheung, "Generation of redundant frame structure for interactive multiview streaming," in *Internat. Packet Video Workshop*, Seattle, WA, USA, May 2009.
- [18] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Trans. on Image Proc.*, vol. 3, no. 3, pp. 744–761, Mar. 2011.
- [19] G. Petrazzuoli, M. Cagnazzo, F. Dufaux, and B. Pesquet-Popescu, "Using distributed source coding and depth image based rendering to improve interactive multiview video access," in *Proc. IEEE Int. Conf. on Image Processing*, Brussels, Belgium, 2011.

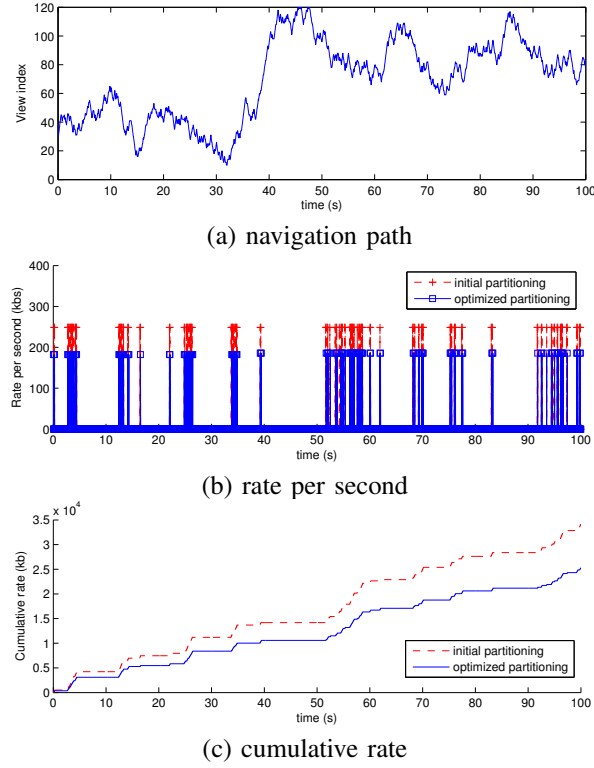


Fig. 19. Rate cost performance with partitioning in $N_V = 3$ navigation segments, *Ballet*, (with the partitioning solutions illustrated in Fig 15).

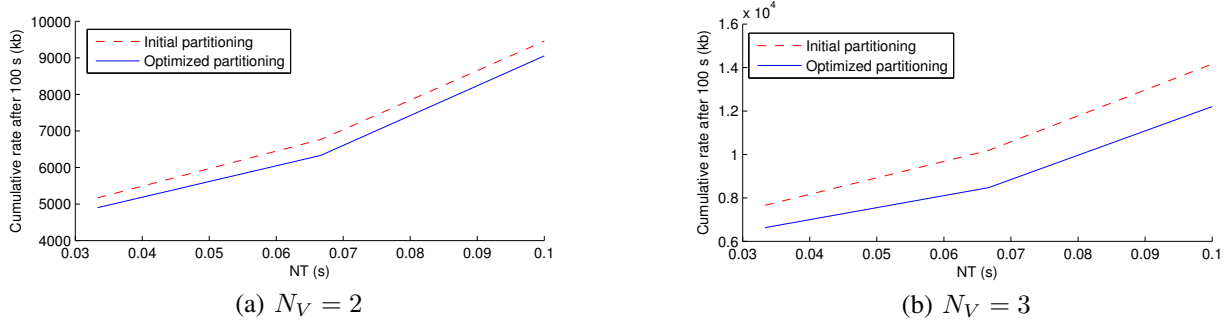


Fig. 20. Averaged cumulative rate after 100s, for 100 navigation paths and for different values of N_T (with the partitioning solutions for *Ballet* sequence illustrated in respectively Figs 14 and 15).

- [20] X. Xiu, G. Cheung, and J. Liang, "Frame structure optimization for interactive multiview video streaming with bounded network delay," in *Proc. IEEE Int. Conf. on Image Processing*, Brussels, Belgium, Sep. 2011.
- [21] N. Cheung, H. Wang, and A. Ortega, "Video compression with flexible playback order based on distributed source coding," in *Proc. SPIE Visual Commun. and Image Processing*, San Jose, California, USA, Nov. 2006.
- [22] X. Xiu, G. Cheung, and J. Liang, "Delay-cognizant interactive streaming of multiview video with free viewpoint synthesis," *IEEE Trans. on Multimedia*, vol. 14, pp. 1109–1126, 2012.
- [23] M. Tanimoto, M. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," *IEEE Signal Processing Magazine*, vol. 11, pp. 67–76, 2011.
- [24] M. Tanimoto, "Ftv: Free-viewpoint television," *IEEE Signal Processing Magazine*, vol. 27, no. 6, pp. 555–570, Jul. 2012.
- [25] T. Maugey, P. Frossard, and G. Cheung, "Consistent view synthesis in interactive multiview imaging," in *Proc. IEEE Int. Conf. on Image Processing*, Orlando, Florida, US, Oct. 2012.
- [26] D. Cohen-Or and A. Kaufman, "Fundamentals of surface voxelization," *Journal of Graphical Models and Image Processing*, vol. 57, pp. 5453–461, 2002.
- [27] D. Tian, P. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3d video," *Proc. of SPIE, the Int. Soc. for Optical Engineering*, vol. 7443, 2009.
- [28] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand, "View synthesis for advanced 3d video systems," *EURASIP J. on Image and Video Proc.*, vol. 2008, 2008.
- [29] [Online]. Available: <http://www.cse.unr.edu/~bebis/CS791E/>
- [30] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. on Image Proc.*, vol. 13, no. 9, pp. 1200–1212, 2004.
- [31] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *IEEE Int. Workshop on Multimedia Sig. Proc.*, Saint Malo, France, Oct. 2010.
- [32] A. Vetro, T. Wiegand, and G. Sullivan, "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 avc standards," *Proc. IEEE*, vol. 99, no. 4, pp. 626–642, Apr. 2011.

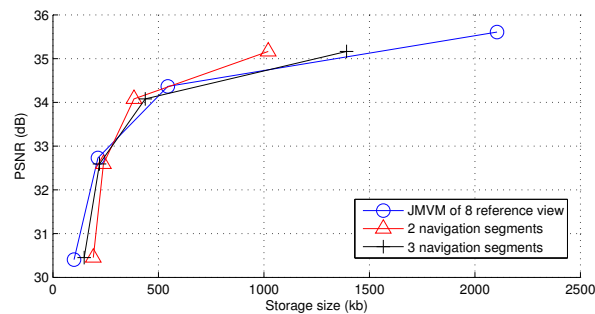


Fig. 21. Distortion of a views 8, 23, 38, 53, 68, 83, 98, 113 as a function of storage size for two proposed partitioning ($N_V = 2$ and $N_V = 3$). It is compared to a solution where the 8 captured reference views are compressed jointly with JMVM (with no auxiliary information).

- [33] C. Fehn, "Depth-image-based rendering (dibr), compression and transmission for a new approach on 3d-tv," *Proc. SPIE, Stereoscopic Image Process. Render.*, vol. 5291, pp. 93–104, 2004.
- [34] J. Shade, S. Gortler, L. He, and R. Szeliski, "Layered depth images," in *Proc. Int. Conf. on Computer graphics and interactive techniques*, New York, NY, USA, Jul. 1998.
- [35] [Online]. Available: <http://research.microsoft.com/en-us/um/people/sbkang/3dvideodownload/>
- [36] A. Gersho and R. Gray, *Vector quantization and signal compression*, R. Gallager, Ed. Kluwer academic publishers, 1992.
- [37] ISO/IEC MPEG & ITU-T VCEG, "Joint multiview video model (JMVM)," Marrakech, Morocco, Jan.13-19 2007.